1,文件存储自定义

- 目的: 能够参考文档,定义出文件存储类
- 操作流程:
  - 1, 定义文件存储类

```python
from django.conf import settings
from django.core.files.storage import Storage

"""
自定义文件存储类:
1, 定义类继承自Storage
2, 必须保证参数能够初始化
3, 必须实现open,save方法

"""
class MyStorage(Storage):
    def __init__(self, base_url=None):
        if not base_url:
            base_url = settings.BASE_URL
        self.base_url = base_url

    def open(self, name, mode='rb'):
        """打开文件的时候调用"""
        pass

    def save(self, name, content, max_length=None):
        """保存文件的时候调用"""
        pass

    def exists(self, name):
        """上传的时候判断图片是否存在了"""
        pass

    def url(self, name):
        """返回图片的url地址"""
        return self.base_url + name
```

  - 2,在settings文件中指定

```
#指定storage的位置
BASE_URL = "http://172.16.12.134:8888/"

#指定自己的文件存储类
DEFAULT_FILE_STORAGE =
'meiduo_mall.utils.fdfs.MyFileStorage.MyStorage
'
```

2.文件存储类测试

- 目的: 能够配置图片访问的域名形式
    - 1修改图片访问的域名形式

```
#etc/hosts
172.16.12.134 image.meiduo.site
```

```
#dev.py
BASE_URL = "http://image.meiduo.site:8888/"
```

    - 2,查看获取图片地址的源代码

3,商品列表页获取

- 目的: 能够编写类视图获取商品列表页
- 操作流程:
    - 1, 根路由(meiduo_mall/urls.py)

```
urlpatterns = [
    ...
    url(r'^',
include('goods.urls',namespace="goods")),
]

```

    - 2,子路由(goods/urls.py)

```
    url(r'^list/(?P<category_id>\d+)/(?P<page_num>\d+)/$',
        views.SkuListView.as_view())

```

- 3,类视图(goods/views.py)

```
from django.shortcuts import render
from django.views import View

class SkuListView(View):
    def get(self,request,category_id,page_num):
        return render(request,'list.html')
```

- 注意点:
  - 注释掉list.html中的{/{ category.id }}

4,商品列表分类信息

- 目的: 能够获取列表页的分类数据
- 操作流程:
  - 1, 封装分类获取方法

```
from goods.models import GoodsChannel


def get_categories():
    # 1,定义字典
    categories = {}

    # 2,查询所有的频道组
    channels = GoodsChannel.objects.order_by('group_id', 'sequence')

    # 3,遍历频道组,组装数据
    for channel in channels:

```

```
        # 3.1 取出组的编号
        group_id = channel.group_id

        # 3.2组装好一个分类的字典
        if group_id not in categories:
            categories[group_id] = {"channels": [], "sub_cats": []}

        # 3.3添加一级分类到channels
        catetory = channel.category
        catetory_dict = {
            "id": catetory.id,
            "name": catetory.name,
            "url": channel.url
        }
        categories[group_id]["channels"].append(catetory_dict)

        # 3.4添加二级分类三级分类
        for cat2 in catetory.subs.all():
            categories[group_id]["sub_cats"].append(cat2)

    # 4,返回分类
    return categories

```

- 2, 调用分类方法

```
class SkuListView(View):
    def get(self,request,category_id,page_num):

        #1,获取分类信息
        categories = get_categories()


        #拼接数据,返回响应
        context = {
            "categories":categories
        }
```

```
12
13          return
    render(request,'list.html',context=context)
14
```

5,商品面包屑导航

- 目的: 能够在商品列表页中显示商品的导航
- 操作流程:
    - 1, 类视图

```
1  class SkuListView(View):
2      def
   get(self,request,category_id,page_num):
3          ...
4
5          #2,获取分类对象
6          category =
   GoodsCategory.objects.get(id=category_id)
7
8          #拼接数据,返回响应
9          context = {
10             "categories":categories,
11             "category":category
12         }
13
14         return
   render(request,'list.html',context=context)
```

    - 2,l模板页面(list.html)

```
  <div class="breadcrumb">
    <a href="http://shouji.jd.com/">{{ category.parent.parent.name }}</a>
    <span>></span>
    <a href="javascript:;">{{ category.parent.name }}</a>
        <span>></span>
    <a href="javascript:;">{{ category.name }}</a>
  </div>
```

6,Paginator分页查询

- 目的: 能够通过Paginator分页获取数据
- 操作流程:
  - 1, 创建对象,获取属性,方法

```
#1, 导入包
In [4]: from django.core.paginator import Paginator



#2, 获取查询区域数据
In [5]: areas = Area.objects.order_by("id")




#3, 创建paginator对象
In [6]: paginator = Paginator(object_list=areas,per_page=10)




#4, 查询第1页的数据
In [7]: page = paginator.page(1)
```

```
12 ------------------------------------------------
   ----------------------------
13 In [9]: page



14 Out[9]: <Page 1 of 323>
15
16 #5,获取当前页的数据
17 In [11]: page.object_list



18 Out[11]: <QuerySet [<Area: 北京市>, <Area: 北京
   市>, <Area: 东城区>, <Area: 西城区>, <Area: 朝阳
   区>, <Area: 丰台区>, <Area: 石景山区>, <Area: 海
   淀区>, <Area: 门头沟区>, <Area: 房山区>]>
19
20 #6,获取当前页
21 In [12]: page.number



22 Out[12]: 1
23
24 #7,获取总页数
25 In [15]: paginator.num_pages



26 Out[15]: 323
27
```

7,列表页分页排序

7,列表页热销排行

8,ES

9,ES安装

10,Haystack

11,Haystack数据索引

12,搜索测试